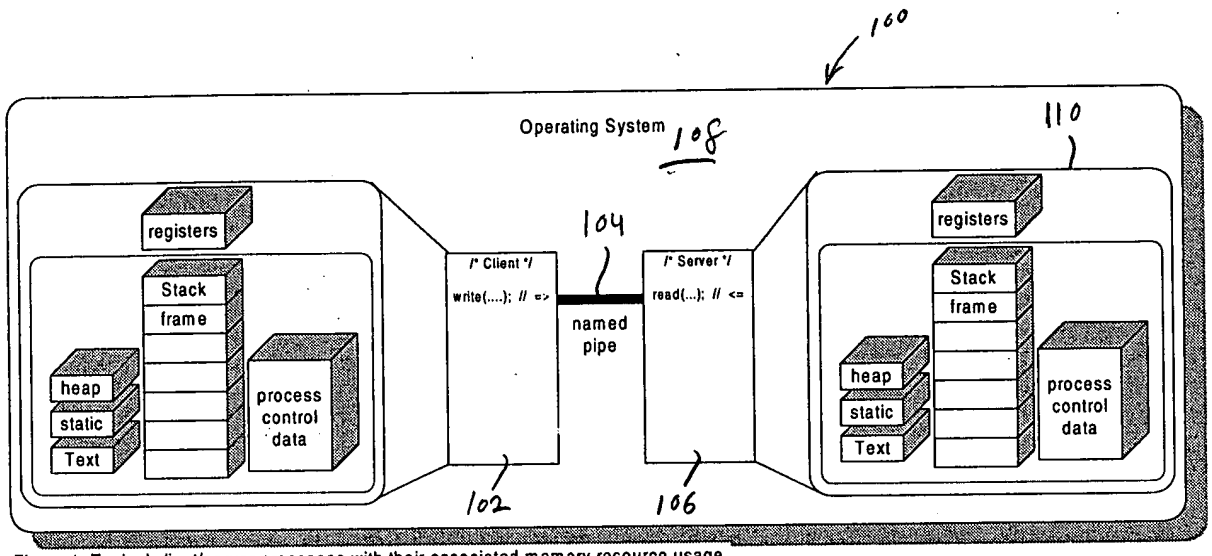

Figure 1. Typical client/server processes with their associated memory resource usage.

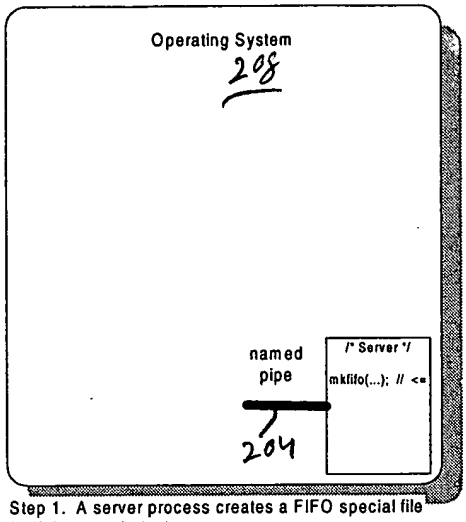

FIG. 2A

Step 1. A server process creates a FIFO special file (A.K.A. named pipe)

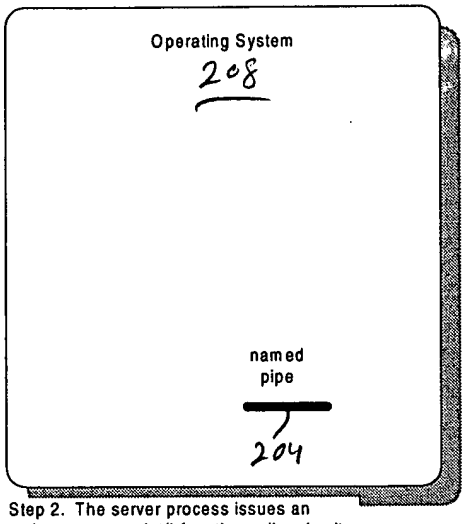

FIG. 2B

Step 2. The server process issues an activate_on_receipt() function call and exits.

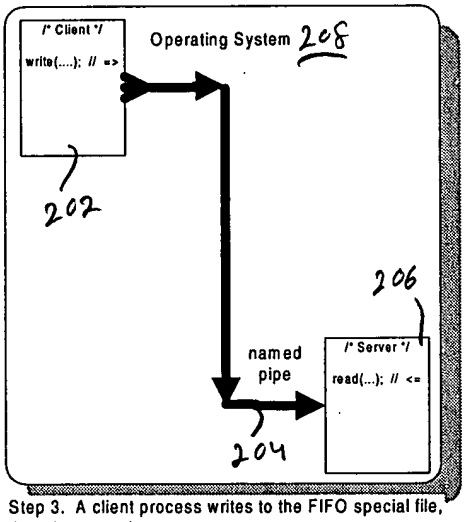

FIG. 2C

Step 3. A client process writes to the FIFO special file, then the operating creates a new server process.

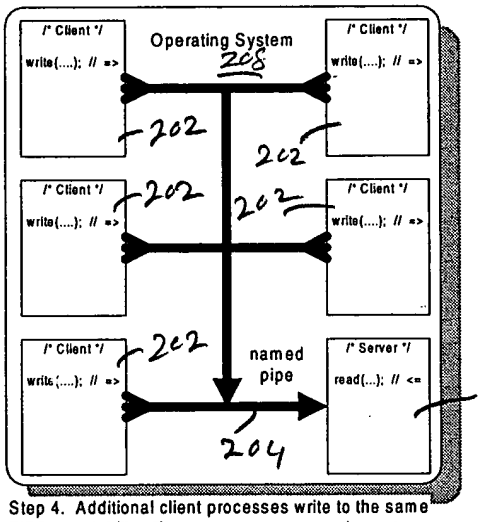

FIG. 2D

Step 4. Additional client processes write to the same FIFO special file. The server process continues to read.

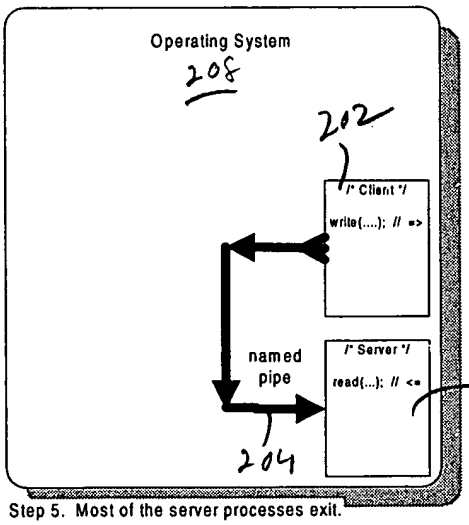

FIG. 2E

Step 5. Most of the server processes exit.

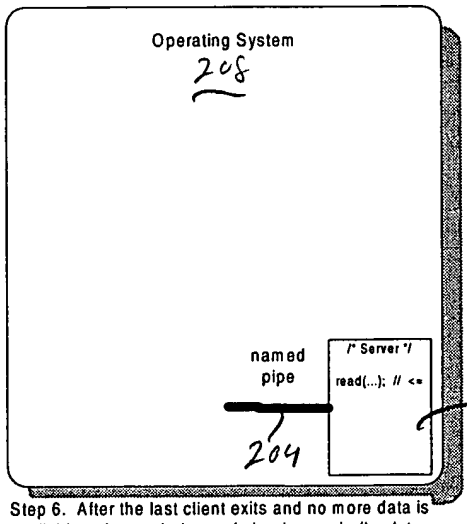

FIG. 2F

Step 6. After the last client exits and no more data is available to be read, the cycle begins again (back to step 2).

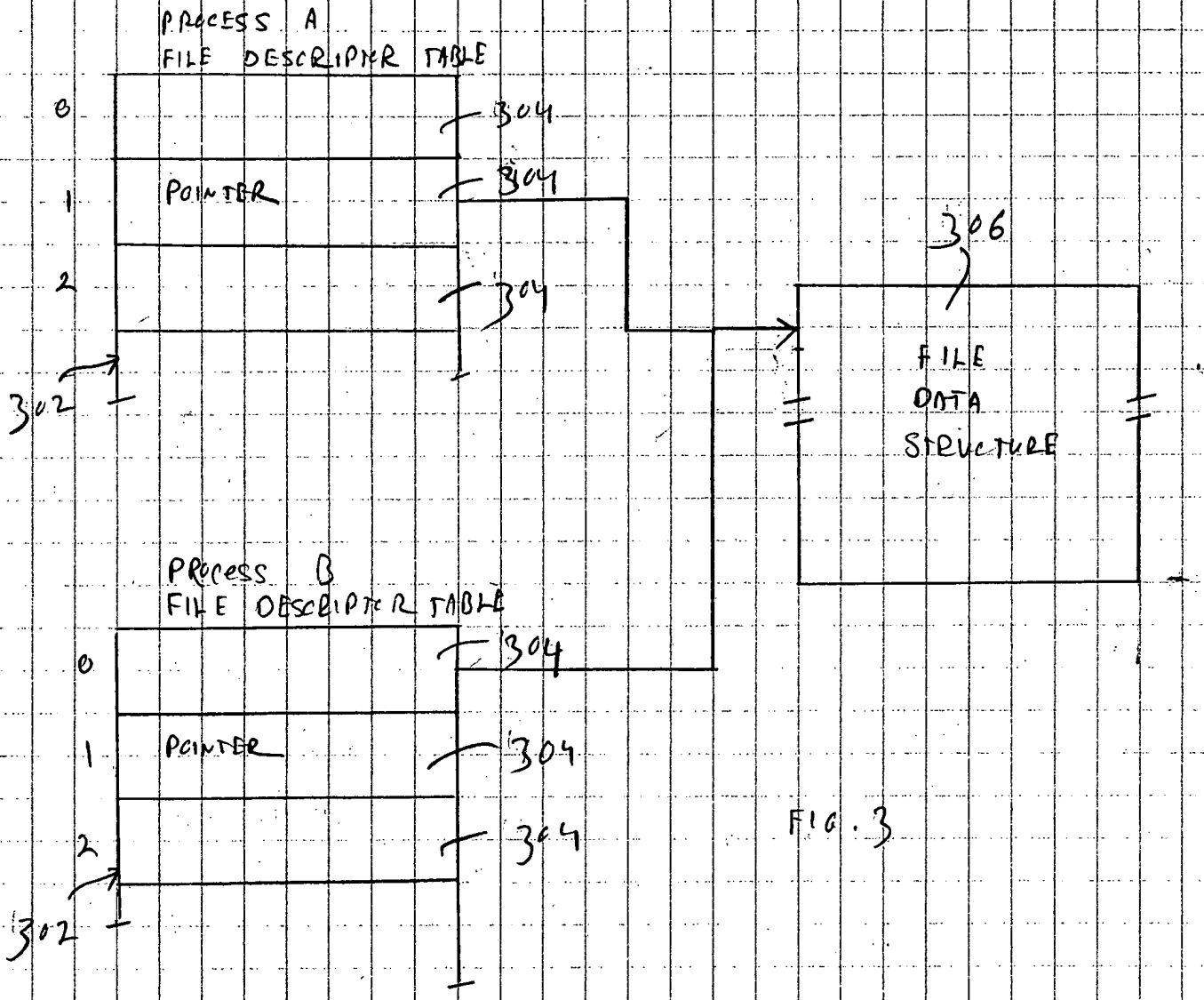
PROCESS A
FILE DESCRIPTER TABLE
0
1
POINTER
2
304
304
304
302
306
FILE
DATA
STRUCTURE
PROCESS B
FILE DESCRIPTER TABLE
0
1
POINTER
2
304
304
304
302

FIG. 3

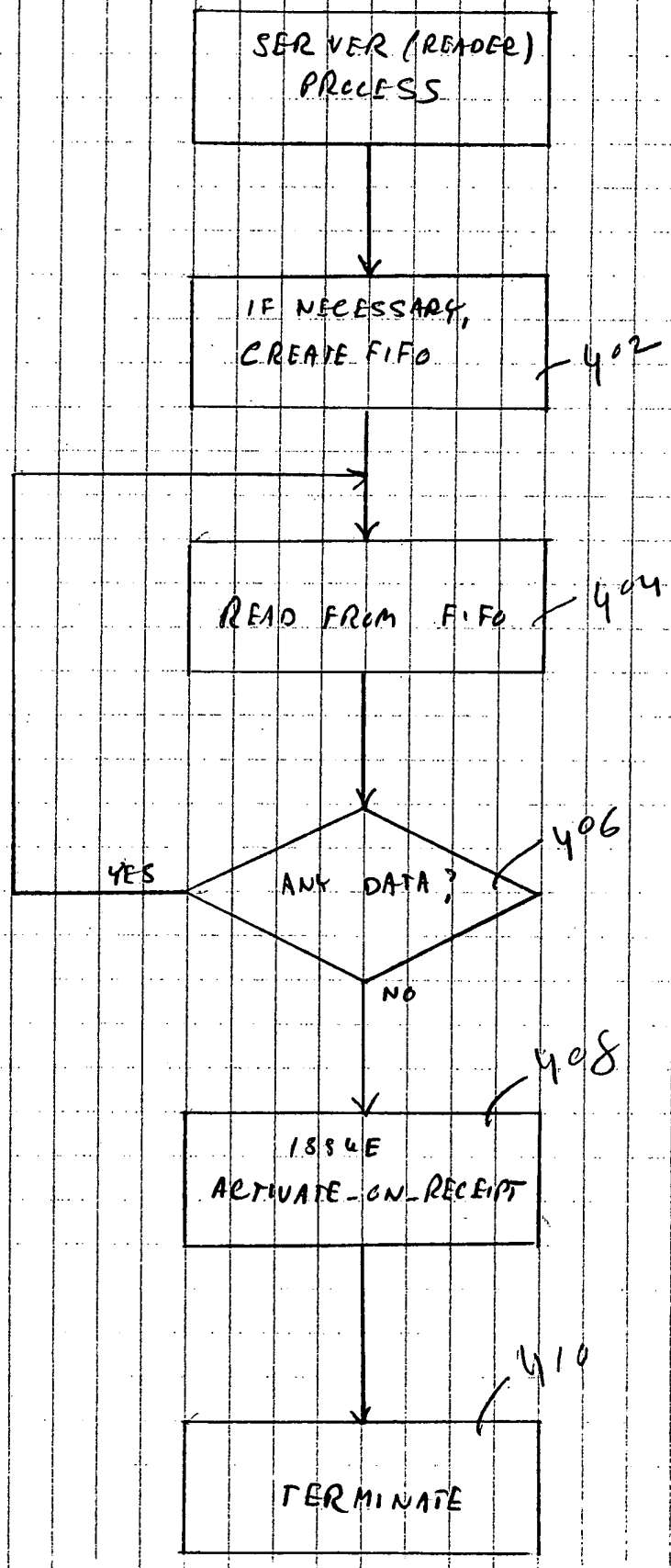
SERVER (READER) PROCESS
IF NECESSARY, CREATE FIFO
402
READ FROM FIFO
404
ANY DATA?
406
YES
NO
ISSUE ACTUATE_ON_RECEIPT
408
TERMINATE
410

FIG. 4

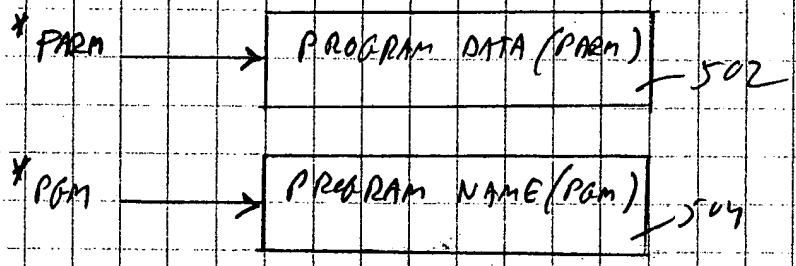

FIG. 5

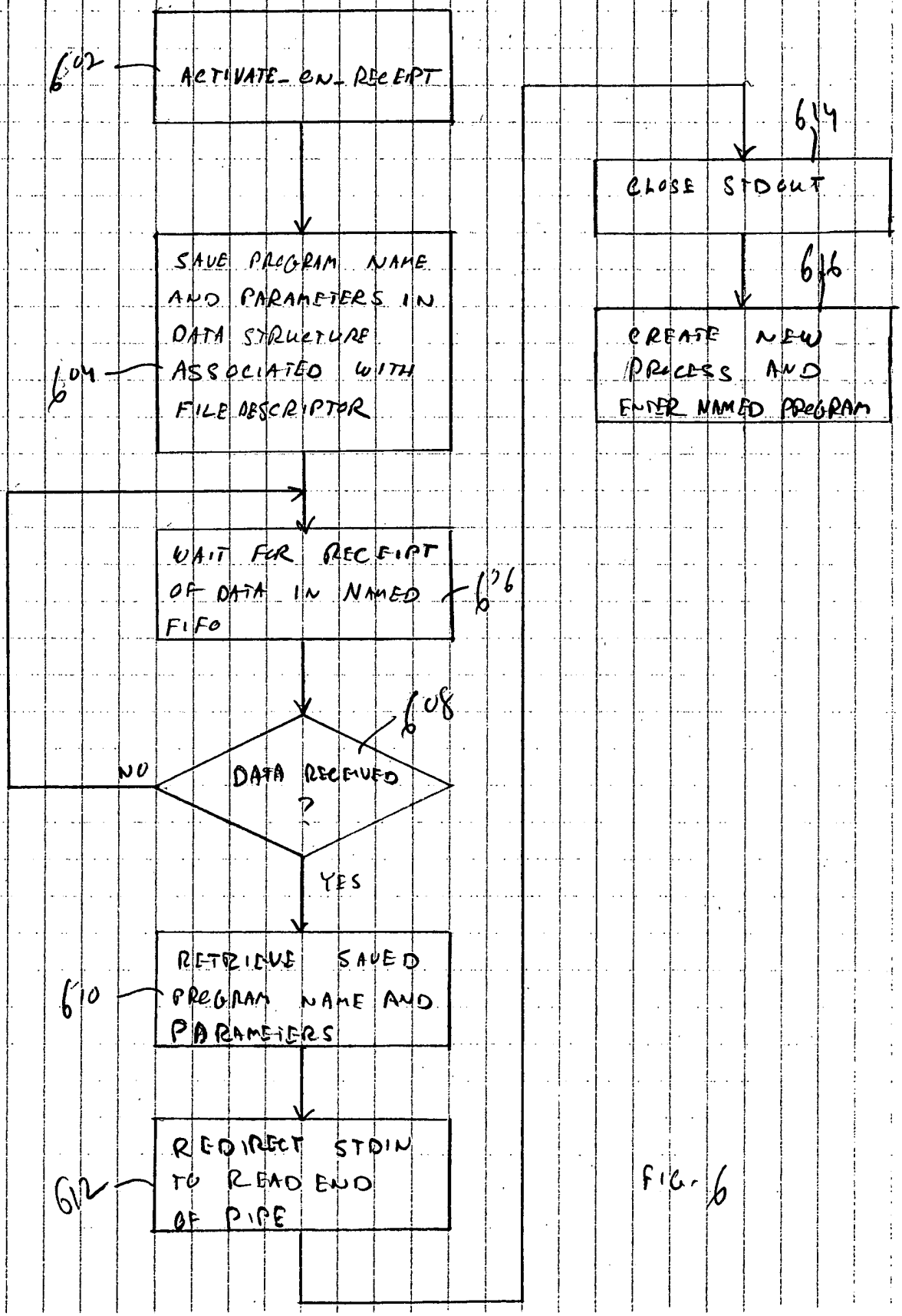
602
ACTIVATE_ON_RECEIPT
604
SAVE PROGRAM NAME AND PARAMETERS IN DATA STRUCTURE ASSOCIATED WITH FILE DESCRIPTOR
606
WAIT FOR RECEIPT OF DATA IN NAMED FIFO
608
DATA RECEIVED ?
NO
YES
610
RETRIEVE SAVED PROGRAM NAME AND PARAMETERS
612
REDIRECT STDIN TO READ END OF PIPE
614
CLOSE STDOUT
616
CREATE NEW PROCESS AND ENTER NAMED PROGRAM

Fig. 6

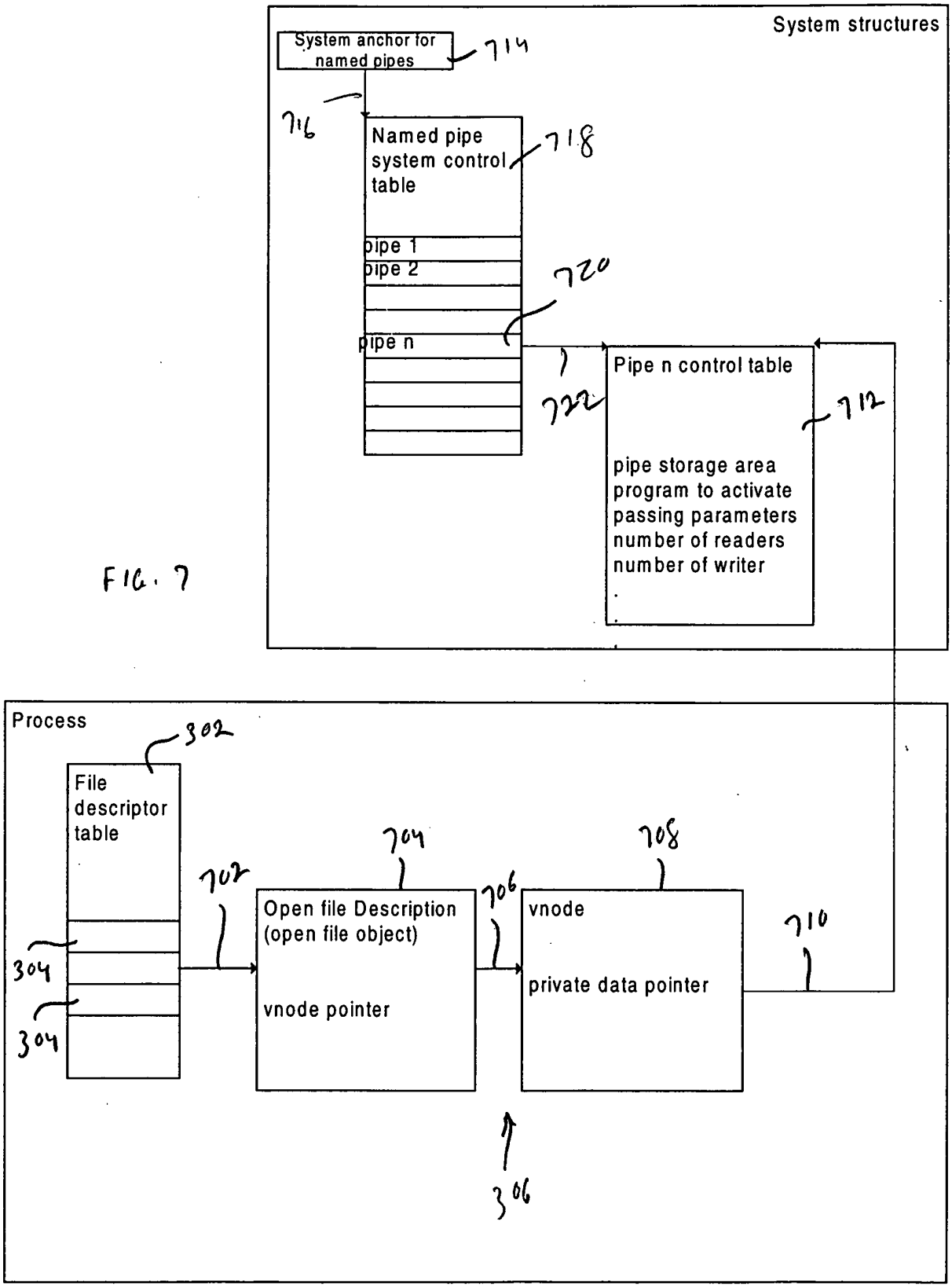
System structures
System anchor for named pipes
714
716
Named pipe system control table
718
pipe 1
pipe 2
720
pipe n
722
Pipe n control table
712
pipe storage area
program to activate
passing parameters
number of readers
number of writer
Process
302
File descriptor table
304
304
702
704
Open file Description (open file object)
vnode pointer
706
708
vnode
private data pointer
710
306

FIG. 7